

# PLCs via Ethernet

## *Improvement over serial*

Thu, Apr 30, 1998

IRMs have been interfaced to PLC hardware using a serial port and DirectNET protocol. The bandwidth of the serial port severely limits obtaining timely (15 Hz) data, although PLC-based data is often not so time-critical as beam data, say. Another option that is now available for connecting to the same PLC hardware is ethernet. This note describes the support for that mode of access, which greatly increases potential throughput.

### **Problem**

The serial port access was supported by a local application, plus additions to the system code. Communication from the system code to the LA to accomplish analog or digital settings is via a message queue. The first time a setting is to be made to a PLC, the message queue is created by the system. The LA itself also acquires the data to update the data pool. The model is simple. A region of memory in the PLC is used for communications. Data acquisition is done by the LA issuing a request for the block of memory that contains all analog and digital words that the PLC provides. A setting is done by sending a message to write into a word of memory. The PLC's ladder logic notices the setting and accomplishes the I/O needed. The same general approach is used for the ethernet mode of access.

Unlike the serial port connection, ethernet has multiple potential targets, each with a different IP address. Various PLC device settings may be delivered to different targets. Also, data acquisition may be collected from different targets.

### **Solution**

Configure multiple instances of a new local application, one for each target PLC crate. Included in the instance parameters is an index number that uniquely identifies that instance and therefore that target. The index number fits in a 4-bit range. Also included is the target IP address. As each LA executes, it acquires data from its "own" PLC interface, depositing the results into the data pool according to its parameters. In addition, it checks its own message queue—called PLCn, where n is the 4-bit index number in hexadecimal—for any settings to be performed. Because the bandwidth is much improved for the ethernet case, it may perform multiple queued settings. This is an improvement over the serial case, in which time is spared to perform only a single setting before performing the next data acquisition.

When a setting is to be made to a PLC-based device, the setting routine checks for the existence of the appropriate message queue according to the index number assigned for that device—either in the analog control field or in the BADDR table entry. If the queue does not exist, then one is created with the appropriate name. Then the setting information is placed into the message queue.

At system reset time, an automatic restore of all settable devices is performed. This occurs before any LAs have been initialized. But the message queue scheme provides a place for the setting information to reside so that it will be noticed when the LA starts up and performs its first data acquisition operation. (This is why the LA does not create the message queue that it waits on.)

**PLC not for beam data**

For PLC data acquisition via the network, or via the serial port, there are delays in receiving the data. The organization of the IRM software is that the Update Task is triggered every 10–15 Hz cycle to update the data pool and fulfill active data requests for which responses are due on that cycle. Local applications are invoked to execute during this time. If a data acquisition request is sent, the response will occur perhaps a few milliseconds later. It must be processed by another task that handles IP communications, but that task will not have a chance to execute until the Update Task completes. By that time, replies will have been delivered to the network so that they cannot include the present cycle's PLC data. Therefore, PLC data in the data pool will be one cycle behind. For this reason, a PLC, or any other data source that supplies its data via the network, should not be used to interface to beam data, in which collecting correlated data is important. The reason that Update Task execution is designed not to be interrupted by other task execution is to insure that network requesters can never see a data pool that is only partially updated.

**LA params**

|          |   |
|----------|---|
| Enable   | LA enable Bit#                                  |
| DataType | Data acq period 00–7F, PLC# 0–E, data type# 0–F |
| ARefAdr  | Analog reference address                        |
| AChans   | #analog words to read                           |
| MapChan  | Mapping channel#                                |
| DRefAdr  | Digital reference address                       |
| DWords   | #digital words to read                          |
| MapBit   | Mapping Bit                                     |
| IPAdrHi  | IP address of target PLC                        |
| IPAdrLo  | "   |

Note that analog and digital data are collected separately, allowing them to reside in two distinct blocks of PLC memory. This facilitates changing their assigned lengths. Example timing tests show that 64 words can be read from PLC memory in 10 ms.